Normalizácia a denormalizácia údajov

Otvorte si databázu s názvom Bookstore_Normalizacia.

Proces rozdelenia údajov do viacerých tabuliek sa nazýva normalizácia údajov. Existuje niekoľko stupňov normalizácie; prvá až tretia fáza sú najjednoduchšie na pochopenie a implementovanie a vo všeobecnosti postačujú pre väčšinu aplikácií. Hoci vyššie úrovne normalizácie sú možné, zvyčajne ich všetci okrem väčšiny skúsených a náročných vývojárov ignorujú.
Na ilustráciu procesu normalizácie použijeme databázu veľkoobchodníka s knihami. Táto databáza musí spracovať nasledujúce informácie:

- ✓ dátumy, kedy boli knihy objednané,
- ✓ zákazníkov, ktorí zadali objednávky,
- ✓ množstvo každej objednanej knihy,
- ✓ názov každej objednanej knihy

Prvá normálna forma

Počiatočná fáza normalizácie, nazývaná prvá normálna forma (1NF), vyžaduje, aby tabuľka dodržiavať nasledujúce pravidlo:

- ✓ Prvá normálna forma stanovuje, že v každom prieniku riadka a stĺpca v tabuľke existuje jedna hodnota a nikdy nie zoznam hodnôt.

Tabuľka má byť dvojrozmerným úložným objektom, v ktorom sa ukladajú viaceré hodnoty v rámci poľa alebo povolenia opakujúcich sa skupín v tabuľke implikuje tretí rozmer údajov. Obrázok 1 zobrazuje prvý pokus o vytvorenie tabuľky na správu objednávok v kníhkupectve (tblBookOrders1).

Obrázok 1

tblBookOrders1

OrderID	OrderDate	Customer	BookTitle
1	5/10/2019	Uptown Books	2 Easy Sushi, 10 Hog Wild Over Ham, 5 Beanie Wienie
2	5/15/2019	Bookmania	3 Crazy Cabbage
3	5/21/2019	Uptown Books	3 New Vegetarian Vegetables, 1 Road Kill Cooking
4	5/25/2019	Jamie's Book Nook	7 Cookie Magic
5	5/30/2019	East Side News	8 Cooking for Twelve, 1 Medieval Meals
6	6/1/2019	Books 'n More	3 Quick Lunches, 3 Quick Dinners, 6 Quick Snacks
7	6/5/2019	Hoopman's	1 Blazing Chicken Recipes, 1 Smokin' Hams
8	6/8/2019	Millie's Book Shop	2 Smokin' Hams
9	6/10/2019	Books 'n More	4 Famous Feeding Frenzies
10	6/11/2019	University Bookshop	3 The Noodle Cookbook, 2 Sizzling Stir Fry
*	(New)		

Record: 1 of 10 | No Filter | Search

Všimnite si, že niektoré kníhkupectvá si objednali viac ako jednu knihu. Hodnota ako 7 pri Cookie Magic v poli BookTitle znamená, že kontakt si objednal sedem kópií kuchárskej knihy s názvom Cookie Magic. Uloženie množstva aj názvu položky do rovnakej bunky je len jedným z niekoľkých spôsobov, ako táto tabuľka porušuje prvú normálnu formu. Tabuľka na obrázku 1 je typická pre plochý typ prístupu k budovaniu databázy. Údaje vo forme plochého typu databázy sú uložené v dvoch dimenziách (riadky a stĺpce) a zanedbávajú tretiu dimenziu (súvisiace tabuľky) v systéme relačnej databázy.
Všimnite si, ako tabuľka na obrázku 1 porušuje prvé pravidlo normalizácie. Mnohé zo záznamov v tejto tabuľke obsahujú viacero hodnôt v poli BookTitle. Napríklad kniha s názvom

Smokin' Hams sa objavuje v záznamoch 7 a 8. Databáza to nijako nedokáže zvládnuť. Mali by sme analyzovať údaje obsiahnuté v poli BookTitle, aby sme zistili, ktoré knihy boli zoradené podľa ktorých kontaktov.
O niečo lepší dizajn je znázornený na obrázku 2 (tblBookOrders2). Množstvo kníh a názvy sú rozdelené do jednotlivých stĺpcov. Každý riadok stále obsahuje všetky údaje pre jednu objednávku. Toto usporiadanie trochu uľahčuje získanie informácie o množstve a názve ale opakujúce sa skupiny pre množstvo a názov (stĺpce Quant1, Title1, Quant2, Title2 a tak ďalej) naďalej porušujú prvé pravidlo normalizácie.

Obrázok 2

tblBookOrders2

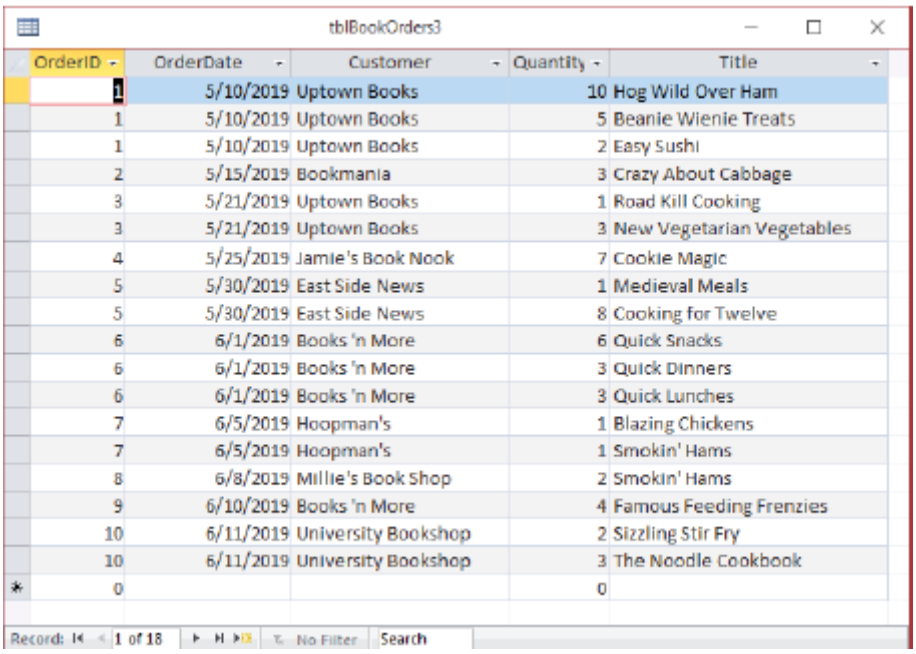
OrderID	OrderDate	Customer	Quant1	Title1	Quant2	Title2	Quant3	Title3	Quant4
1	5/10/2019	Uptown Books	2	Easy Sushi	10	Hog Wild Over Ham	5	Beanie Wienie	
2	5/15/2019	Bookmania	3	Crazy Cabbage					
3	5/21/2019	Uptown Books	3	New Vegetarian Vegetables	1	Road Kill Cooking			
4	5/25/2019	Jamie's Book Nook	7	Cookie Magic					
5	5/30/2019	East Side News	8	Cooking for Twelve	1	Medieval Meals			
6	6/1/2019	Books 'n More	3	Quick Lunches	3	Quick Dinners	6	Quick Snacks	
7	6/5/2019	Hoopman's	1	Blazing Chicken Recipes	1	Smokin' Hams			
8	6/8/2019	Millie's Book Shop	2	Smokin' Hams					
9	6/10/2019	Books 'n More	4	Famous Feeding Frenzies					
10	6/11/2019	University Bookshop	3	The Noodle Cookbook	2	Sizzling Stir Fry			
*	(New)		0		0		0		0

Record: 1 of 10 | No Filter | Search

Dizajn na obrázku 2 je stále dosť nemotorný a ťažko sa s ním pracuje. Stĺpce, ktoré obsahujú informácie o Množstve a Názve sú stálymi poľami tabuľky. Vývojár musí pridať dostatok stĺpcov na umiestnenie maximálneho počtu kníh, ktoré je možné zakúpiť na jednu objednávku. Predpokladajme napríklad, že vývojár predpokladá, že žiadne kníhkupectvo si nikdy neobjedná viac ako 50 kníh naraz. To znamená, že sa pridá 100 stĺpcov do tabuľky (pre každý objednaný titul knihy sú potrebné dva stĺpce – Množstvo a Názov). Ak si kníhkupectvo objedná jednu knihu, 98 stĺpcov by zostalo prázdnych v tabuľke, čo je veľmi neefektívna situácia.

Na základe návrhu znázorneného na obrázku 2 by bolo mimoriadne ťažké získavať informácie o predaji konkrétnej knihy z tabuľky tblBookOrders2. Množstvo predané za každú knihu je roztrúsené po celej tabuľke, v rôznych riadkoch a rôznych stĺpcoch, čo sťažuje prácu a získanie údajov o predaji knihy. Taktiež, ak objednávka kníh presahuje 50 kníh, tabuľka musí byť prerobená, aby sa do nej zmestili ďalšie stĺpce potrebné pre objednávku. Samozrejme, používateľ môže pridať druhý riadok pre objednávku, čo sťažuje prácu s údajmi v tabuľke tak, ako bolo pôvodne zamýšľané.
Obrázok 3 zobrazuje tblBookOrders3, novú tabuľku vytvorenú z údajov na obrázku 2. Namiesto skladania viacerých objednávok kníh do jedného záznamu v tblBookOrders3 každý záznam obsahuje jednu knihu objednanú zákazníkom. Je síce viac záznamov, ale je to potrebné a narába sa oveľa jednoduchšie. Prvá normálna forma je oveľa efektívnejšia, pretože tabuľka neobsahuje žiadne nepoužité polia. Každé pole má význam pre tabuľku.

Obrázok 3



tblBookOrders3

OrderID	OrderDate	Customer	Quantity	Title
1	5/10/2019	Uptown Books	10	Hog Wild Over Ham
1	5/10/2019	Uptown Books	5	Beanie Wienie Treats
1	5/10/2019	Uptown Books	2	Easy Sushi
2	5/15/2019	Bookmania	3	Crazy About Cabbage
3	5/21/2019	Uptown Books	1	Road Kill Cooking
3	5/21/2019	Uptown Books	3	New Vegetarian Vegetables
4	5/25/2019	Jamie's Book Nook	7	Cookie Magic
5	5/30/2019	East Side News	1	Medieval Meals
5	5/30/2019	East Side News	8	Cooking for Twelve
6	6/1/2019	Books 'n More	6	Quick Snacks
6	6/1/2019	Books 'n More	3	Quick Dinners
6	6/1/2019	Books 'n More	3	Quick Lunches
7	6/5/2019	Hoopman's	1	Blazing Chickens
7	6/5/2019	Hoopman's	1	Smokin' Hams
8	6/8/2019	Millie's Book Shop	2	Smokin' Hams
9	6/10/2019	Books 'n More	4	Famous Feeding Frenzies
10	6/11/2019	University Bookshop	2	Sizzling Stir Fry
10	6/11/2019	University Bookshop	3	The Noodle Cookbook
0			0	

Record: 1 of 18 No Filter Search

Tabuľka na obrázku 3 obsahuje rovnaké údaje ako na obrázku 1 a 2. Nové usporiadanie však výrazne uľahčuje prácu s údajmi. Napríklad dotazy sa dajú jednoducho zostaviť tak, aby vrátili celkový počet konkrétnej knihy objednanej zákazníkmi alebo určili, ktoré tituly si objednalo konkrétne kníhkupectvo.
Optimalizácia dizajnu tabuľky však v tomto bode nie je dokončená. S údajmi BookOrders a ostatnými tabuľkami v tejto aplikácii je potrebné ešte veľa urobiť. Najmä tabuľka zobrazená na obrázku 3 obsahuje redundantné informácie. Názvy kníh sa opakujú vždy, keď si zákazníci objednajú rovnakú knihu, a číslo objednávky a dátum objednávky sa zopakujú pre všetky riadky objednávky.
Ďalším problémom je skutočnosť, že OrderID už nemožno použiť ako hlavný kľúč tabuľky. Keďže OrderID je duplikované pre každý titul knihy v objednávke, nemožno ho použiť na identifikáciu jednotlivých záznamov v tabuľke. Namiesto toho je pole OrderID teraz kľúčovým poľom pre tabuľku a možno ho použiť na vyhľadanie všetkých záznamov relevantných pre konkrétnu objednávku. Ďalší krok optimalizácie túto situáciu napraví.

Druhá normálna forma

Efektívnejší dizajn je výsledkom rozdelenia údajov v tblBookOrders do viacerých tabuliek na dosiahnutie druhej normálnej formy (2NF). Druhé pravidlo normalizácie hovorí:

- ✓ Údaje, ktoré nie sú priamo závislé od hlavného kľúča tabuľky, sa presunú do inej tabuľky.

Toto pravidlo znamená, že tabuľka by mala obsahovať údaje, ktoré predstavujú jednu entitu. Pretože postupne meníme jednu nenormalizovanú tabuľku na normalizované údaje, tblBookOrders3 nemá hlavný kľúč. Túto skutočnosť budeme zatiaľ ignorovať a každý riadok v tabuľke budeme považovať za entitu. Všetky údaje v tomto riadku, ktoré nie sú integrálnou súčasťou entity, sa presunú do inej tabuľky. V tblBookOrders3 nie je pole Zákazník ani pole Názov neoddeliteľnou súčasťou objednávky a mali by byť presunuté do inej tabuľky.

Identifikácia entít

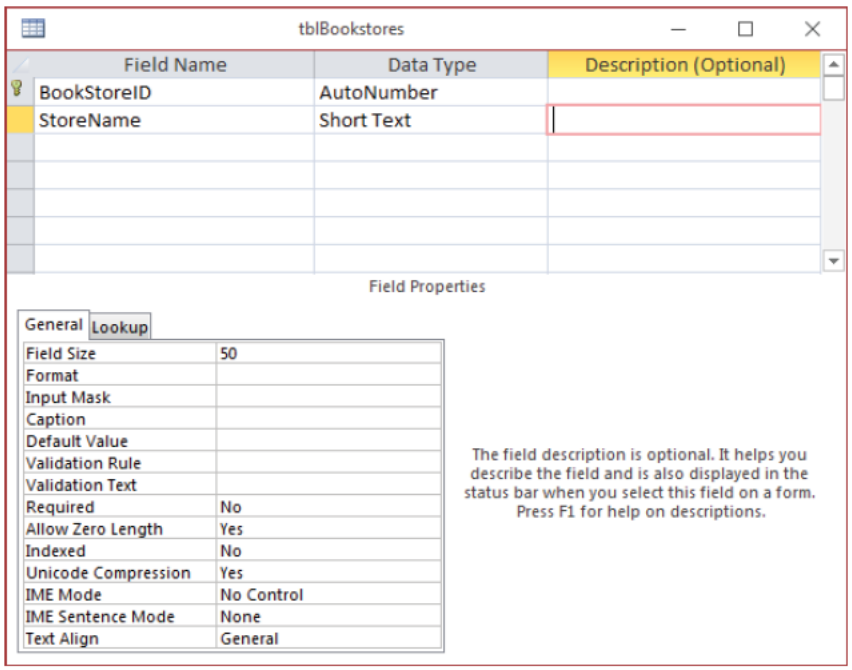
Nie sú však zákazníci neoddeliteľnou súčasťou objednávky? Áno sú. Údaje, ktoré sú uložené v tblBookOrders3 v poli Zákazník, je však meno zákazníka. Ak by si zákazník zmenil mená, zásadne by to nezmenilo poradie. Podobne, zatiaľ čo kniha je neoddeliteľnou súčasťou objednávky, názov knihy nie je.

Na nápravu tejto situácie potrebujeme samostatné tabuľky pre zákazníkov a knihy. Najprv vytvorte novú tabuľku s názvom tblBookstores, ako je znázornené na obrázku 4.

Pre vytvorenie tabuľky tblBookstores, postupujte takto:

1) Kliknite na kartu Vytvoriť a na Návrh tabuľky.
2) Pridajte pole s názvom BookstoreID s typom údajom Automatické číslovanie.
3) Priraďte tomuto poľu zároveň hlavný kľúč.
4) Pridajte ďalšie pole pomenované ako StoreName s typom údajov Krátky text.
5) Nastavte tomuto poľu veľkosť 50.
6) Uložte tabuľku pod názvom tblBookstores.

Obrázok 4



Vieme si predstaviť, že chceme uchovávať nejaké ďalšie informácie o zákazníkoch, ako sú ich poštové adresy a telefónne čísla. V súčasnosti dostávame naše údaje do 2NF presunutím údajov, ktoré nie sú súčasťou objednávky, do vlastnej tabuľky.

Teraz vytvorte tabuľku pre knihy:

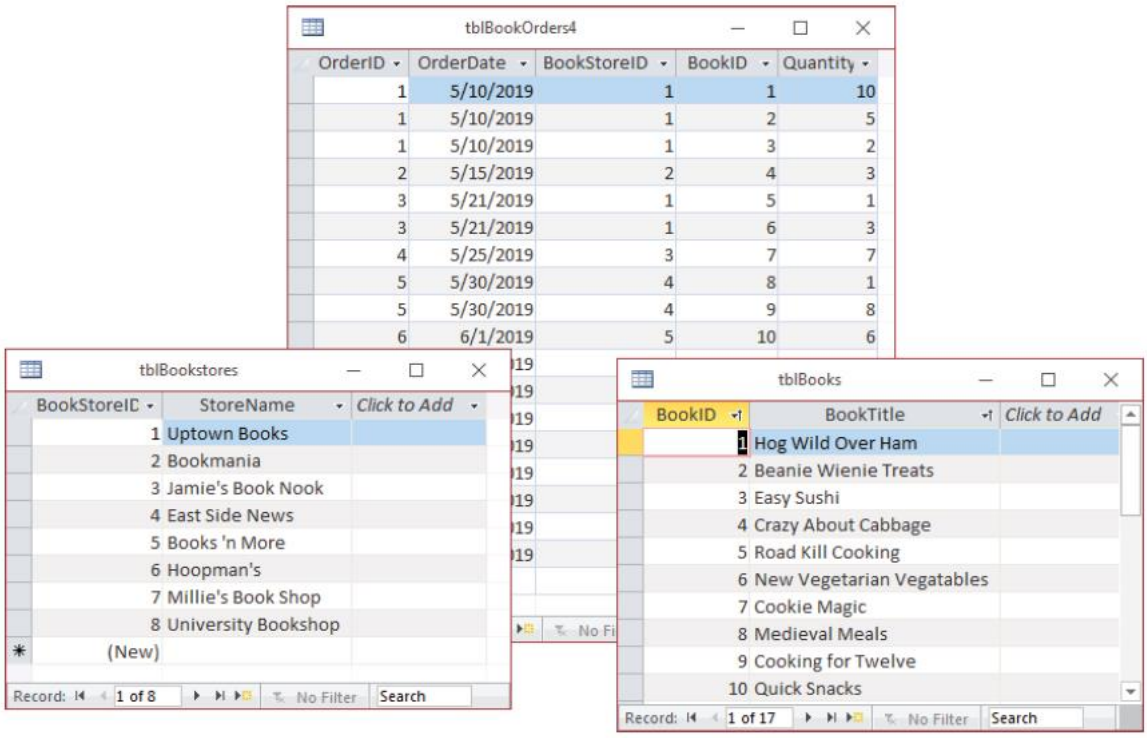
1) Kliknite na kartu Vytvoriť a na Návrh tabuľky.
2) Vytvorte pole BookID s typom údajov Automatické číslovanie.
3) Pridajte tomuto poľu aj hlavný kľúč.
4) Pridajte pole s názvom BookTitle s typom údajov Krátky text.
5) Uložte tabuľku pod názvom tblBooks.

Zákazník a kniha sú stále neoddeliteľnou súčasťou objednávky (iba nie meno a názov) a potrebujeme spôsob, ako navzájom spojiť tabuľky. Zatiaľ čo zákazník môže zmeniť meno,

nemôže zmeniť BookstoreID, pretože sme ho vytvorili my a máme ho pod kontrolou. Podobne môže vydavateľ zmeniť názov knihy, ale nie BookID. Hlavné kľúče tblBookstores a tblBooks sú spoľahlivým zabezpečením pre objekty, ktoré identifikujú, bez ohľadu na to, aké ďalšie informácie sa môžu zmeniť.

Obrázok 5 zobrazuje naše tri tabuľky, ale namiesto mena zákazníka a názvu knihy teraz tblBookOrders4 obsahuje hlavný kľúč súvisiaceho záznamu v tblBookstores aj tblBooks. Keď sa hlavný kľúč jednej tabuľky používa ako pole v inej tabuľke, nazýva sa cudzí kľúč.

Obrázok 5



Pred rozdelením údajov o zákazníkoch do vlastnej tabuľky, ak by Uptown Books zmenil svoj názov na Uptown Books and Periodicals, museli by sme identifikovať všetky riadky v tblBookOrders3, ktoré mali zákazníka Uptown Books, a zmeniť hodnotu poľa pre každý identifikovaný riadok.
Prehliadnutie inštancie mena zákazníka počas tohto procesu sa nazýva anomália aktualizácie a výsledkom sú záznamy, ktoré nie sú konzistentné s ostatnými záznamami v databáze. Z pohľadu databázy sú Uptown Books a Uptown Books and Periodicals dve úplne odlišné organizácie, aj keď vieme, že ide o ten istý obchod. V dotaze na získanie všetkých objednávok zadaných Uptown Books and Periodicals budú chýbať všetky záznamy, ktoré ešte majú Uptown Books v poli Customer kvôli anomálii aktualizácie.

Ďalšou výhodou odstránenia mena zákazníka z tabuľky objednávok je, že názov teraz existuje iba na jednom mieste v databáze. Ak Uptown Books zmení svoj názov na Uptown Books and Periodicals, teraz musíme zmeniť iba jeho záznam v tabuľke tblBookstores. Táto jediná zmena sa prejaví v celej databáze vrátane všetkých formulárov a zostáv, ktoré používajú informácie o mene zákazníka.

Identifikácia samostatných entít a vloženie ich údajov do samostatných tabuliek je skvelým prvým krokom k dosiahnutiu druhej normálnej formy. Ale ešte sme neskončili. Naša tabuľka objednávok stále nemá jedinečné pole, ktoré by sme mohli použiť ako primárny kľúč. Pole OrderID má opakujúce sa hodnoty, ktoré poskytujú vodítko, že na dosiahnutie 2NF je potrebné vykonať ešte viac práce.

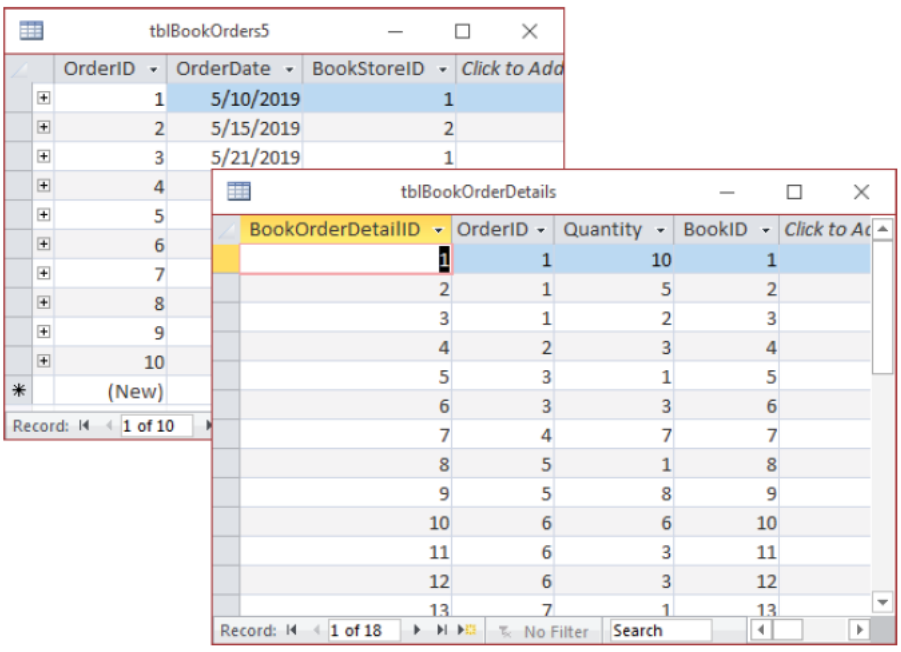
Menej zjavné entity

Zákazníci a knihy sú fyzické objekty, ktoré sa dajú ľahko identifikovať ako samostatné entity. Ďalší krok je trochu abstraktnejší. Naša tabuľka objednávok, teraz nazývaná tblBookOrders4, stále obsahuje informácie o dvoch samostatných, ale súvisiacich entitách. Objednávka je jedna entita a podrobnosti objednávky (jednotlivé riadky objednávky) sú všetky samostatné entity.

Prvé tri záznamy tblBookOrders4 zobrazené na obrázku 5 obsahujú rovnaké OrderID, OrderDate a BookStoreID. Tieto tri polia sú charakteristikami objednávky ako celku, nie každého jednotlivého riadku objednávky. Polia Quantity a BookID obsahujú rôzne hodnoty v týchto troch prvých záznamoch. Quantity a BookID sú charakteristiky konkrétneho riadku na objednávke.

Posledným krokom k tomu, aby sa údaje o našej objednávke dostali do druhej normálnej formy, je vloženie informácií, ktoré sú neoddeliteľnou súčasťou objednávky ako celku, do samostatnej tabuľky pre každý riadok objednávky. Vytvorte novú tabuľku s názvom tblBookOrderDetails s poľami BookOrderDetailID, Quantity a BookID. BookOrderDetailID je pole s typom údajov Automatické číslovanie, ktoré bude slúžiť ako hlavný kľúč, a BookID je pole cudzieho kľúča, ktoré používame na spojenie týchto dvoch tabuliek. Obrázok 6 zobrazuje našu novú tabuľku objednávok, tblBookOrders5, a našu novú tabuľku podrobností, tblBookOrderDetails.

Obrázok 6



Pole OrderID v tblBookOrders5 bolo odstránené a bolo vytvorené nové pole s typom Automatzické číslovanie s názvom OrderID. Teraz, keď máme jedinečné pole v tabuľke objednávok, môžeme nastaviť OrderID ako hlavný kľúč. Všetky údaje v každom zázname tblBookOrders5 priamo súvisia s entitou objednávky. Alebo v jazyku 2NF sú všetky údaje priamo závislé od hlavného kľúča.
Pole OrderID v tblBookOrderDetails je cudzí kľúč, ktorý sa používa na spojenie týchto dvoch tabuliek. Obrázok 6 ukazuje, že prvé tri záznamy v tblBookOrderDetails zobrazujú OrderID 1, ktoré sa mapuje na prvý záznam tblBookOrders5.
Všetky polia v tblBookOrderDetails sú priamo závislé od hlavného kľúča BookOrderDetailID. Množstvo z prvého záznamu, 10, priamo súvisí s touto riadkovou položkou v objednávke. Na zákazku ako celok sa vzťahuje len nepriamo, rovnako ako množstvá z nasledujúcich dvoch záznamov 5 a 2. Tento nepriamy vzťah je vytvorený zahrnutím cudzieho kľúča OrderID do záznamu.
Pôvodná tabuľka tblBookOrders1 obsahovala údaje o niekoľkých rôznych entitách v každom zázname. Prostredníctvom série krokov rozdelíme údaje do štyroch tabuliek – tblBookOrders5, tblBookOrderDetails, tblBookstores a tblBooks – z ktorých každá obsahuje údaje o jednej entite. Naše dáta sú konečne v druhej normálnej forme.

Rozdelenie tabuľky na jednotlivé tabuľky, z ktorých každá popisuje nejaký aspekt údajov, sa nazýva dekompozícia. Rozklad je veľmi dôležitou súčasťou procesu normalizácie. Aj keď sa tabuľky zdajú menšie ako pôvodná tabuľka (pozri obrázok 1), údaje obsiahnuté v tabuľkách sú rovnaké ako predtým.
Vývojár, ktorý pracuje s tabuľkami kníhkupectva, dokáže pomocou dotazov opätovne kombinovať údaje v štyroch tabuľkách novými a zaujímavými spôsobmi. Bolo by celkom jednoduché určiť, koľko kníh každého typu si objednali rôzni zákazníci alebo koľkokrát bola konkrétna kniha objednaná. Keď sa spojí s tabuľkou obsahujúcou informácie, ako sú jednotkové náklady knihy, predajná cena knihy atď., vyjasní sa dôležitý finančný stav obchodníka s knihami.
Všimnite si tiež, že počet záznamov v tblBookOrders5 bol znížený. Toto je jedna z niekoľkých výhod používania relačnej databázy. Každá tabuľka obsahuje iba toľko údajov, koľko je potrebné na reprezentáciu entity (v tomto prípade účtovnej objednávky) opísanej tabuľkou. Je to oveľa efektívnejšie ako pridávanie duplicitných hodnôt polí (pozrite si obrázok 1) pre každý nový záznam pridaný do tabuľky.

Porušovanie pravidiel
Z času na čas možno zistíte, že je potrebné porušiť pravidlá. Predpokladajme napríklad, že kníhkupectvá majú nárok na zľavy na základe objemu nákupov za posledný rok. Pri prísnom dodržiavaní pravidiel normalizácie by percento zľavy malo byť zahrnuté v tabuľke tblBookstores. Zľava totiž závisí od zákazníka, nie od objednávky.
Ale možno je zľava aplikovaná na každú objednávku do istej miery svojvoľná. Možno, že obchodník s knihami povolí predajcom znížiť špeciálne ponuky pre vážených zákazníkov. V tomto prípade možno budete chcieť zahrnúť stĺpec Zľava do tabuľky obsahujúcej informácie o objednávkach kníh, aj keď to znamená duplikovanie informácií v mnohých záznamoch. Môžete uložiť tradičnú zľavu ako súčasť záznamu zákazníka v tblBookstores a použiť ju ako predvolenú hodnotu pre stĺpec Zľava, ale povoliť predajcovi, aby prepísal hodnotu zľavy, keď sa so zákazníkom dohodne.

V skutočnosti sa len zdá, že to porušuje druhú normálnu formu. Predvolená zľava je priamo závislá od zákazníka. Skutočná poskytnutá zľava je priamo závislá od objednávky. Podobná situácia môže nastať s dodacími adresami. Zákazník si môže nechať poslať väčšinu svojich objednávok, ale príležitostne môže chcieť, aby bola objednávka odoslaná priamo zákazníkovi. Dodacia adresa zákazníka sa priamo vzťahuje na zákazníka a adresa, na ktorú bola objednávka skutočne odoslaná, priamo súvisí s objednávkou. Hodnoty v tabuľkách objektov, ktoré slúžia ako predvolené hodnoty v tabuľkách transakcií, sú bežné vo veľkých databázach.

Tretia normálna forma

Posledný krok normalizácie, nazývaný tretia normálna forma (3NF), vyžaduje odstránenie všetkých polí, ktoré možno odvodiť z údajov obsiahnutých v iných poliach tabuľky alebo iných tabuliek v databáze. Povedzme napríklad, že manažér predaja trvá na tom, aby ste pridali pole, ktoré bude obsahovať celkový počet kníh v objednávke v tabuľke Objednávky. Tieto informácie by sa samozrejme vypočítali z poľa Množstvo v tblBookOrderDetails.
V skutočnosti nie je potrebné pridávať nové pole OrderTotal do tabuľky Orders. Access jednoducho vypočíta túto hodnotu z údajov, ktoré sú dostupné v databáze. Jedinou výhodou ukladania súčtov objednávok ako súčasti databázy je ušetriť niekoľko milisekúnd potrebných na to, aby Access načítal a vypočítal informácie, keď vypočítané údaje potrebuje formulár alebo zostava.
Odstránenie vypočítaných údajov zachová integritu údajov vo vašej databáze. Obrázok 6 zobrazuje tri záznamy v tblBookOrderDetails, ktoré súvisia s objednávkou s OrderID 1. Po sčítaní poľa Quantity môžete vidieť, že bolo objednaných 22 kníh (nie je vidno všetky záznamy na obrázku). Ak by existovalo pole OrderTotal a súčet by bol nesprávne zadaný ako 33 namiesto 22, údaje by boli nekonzistentné. V zostave zobrazujúcej celkový počet kníh objednaných pomocou poľa OrderTotal by sa zobrazilo iné číslo ako v zostave založenej na tabuľke Podrobnosti.
V závislosti od aplikácií, ktoré vytvoríte, môžete nájsť dobré dôvody na ukladanie vypočítaných údajov do tabuliek, najmä ak je vykonávanie výpočtov zdĺhavým procesom alebo ak je uložená hodnota potrebná ako kontrola vypočítanej hodnoty vytlačenej v správach. Môže byť efektívnejšie vykonávať výpočty počas zadávania údajov (keď sa údaje spracúvajú po jednom zázname) namiesto pri tlači zostáv (keď sa manipuluje s mnohými tisíckami záznamov na vytvorenie jednej zostavy).
Ako sa dočítate v časti „Denormalizácia“ ďalej v tejto kapitole, existuje niekoľko dobrých dôvodov, prečo by ste sa mohli rozhodnúť zahrnúť vypočítané polia do databázovej tabuľky. Rozhodnutie o denormalizácii je najčastejšie založené na potrebe uistiť sa, že v databáze je uložená rovnaká vypočítaná hodnota, aká je vytlačená v správe.

Viac o anomáliách

Túto záležitosť týkajúcu sa anomálií aktualizácií je dôležité mať na pamäti. Celý účel normalizácie tabuliek vo vašich databázach je dosiahnuť maximálny výkon s minimálnym úsilím o údržbu.
Z nenormalizovaného návrhu databázy sa môžu vyskytnúť tri typy chýb. Dodržiavanie pravidiel uvedených v tejto kapitole vám pomôže vyhnúť sa nasledujúcim chybám:

- ✓ Anomália vkladania - pri pridávaní nového záznamu do inej tabuľky sa v súvisiacej tabuľke vyskytne chyba. Povedzme napríklad, že ste pridali pole OrderTotal opísané v predchádzajúcej časti. Po spracovaní objednávky zákazník zavolá a zmení počet objednaných kníh alebo pridá nový knižný titul k tej istej objednávke. Pokiaľ ste

databázu starostlivo nenavrhli tak, aby automaticky aktualizovala vypočítané pole OrderTotal, údaje v tomto poli budú pri vkladaní nových údajov do tabuľky chybné.

- ✓ Anomália vymazania - spôsobí náhodnú stratu údajov pri vymazaní záznamu z tabuľky. Predpokladajme, že tabuľka tblBookOrders3 obsahuje názov, adresu a ďalšie kontaktné informácie pre každé kníhkupectvo. Odstránenie posledného zostávajúceho záznamu obsahujúceho objednávku konkrétneho zákazníka spôsobí neúmyselnú stratu kontaktných informácií zákazníka. Uchovávanie kontaktných informácií zákazníka v samostatnej tabuľke uchováva a chráni tieto údaje pred náhodnou stratou. Vyhýbanie sa anomáliám pri odstraňovaní je jedným z dobrých dôvodov, prečo nepoužívať kaskádové vymazávanie v tabuľkách.
- ✓ Aktualizácia anomálie - ukladanie údajov, ktoré nie sú závislé od hlavného kľúča tabuľky, spôsobuje, že pri každej zmene nezávislých informácií musíte aktualizovať viacero riadkov. Uchovávanie nezávislých údajov (ako sú informácie o kníhkupectve) v ich vlastnej tabuľke znamená, že je potrebné aktualizovať iba jednu inštanciu informácií.

Denormalizácia

Po ukázaní všetkých dôvodov, prečo je normalizácia databáz dobrý nápad, zvážte, kedy by ste sa mohli zámerne rozhodnúť denormalizovať tabuľky alebo použiť nenormalizované tabuľky.
Všeobecne povedané, normalizujete údaje v snahe zlepšiť výkon vašej databázy. Napríklad, napriek všetkému vášmu úsiliu budú niektoré vyhľadávania časovo náročné. Dokonca aj pri použití starostlivo indexovaných a normalizovaných tabuliek si niektoré vyhľadávania vyžadujú dosť času, najmä ak sú vyhľadávané údaje komplikované alebo je ich veľké množstvo.
Podobne môže výpočet niektorých vypočítaných hodnôt trvať dlho. Možno zistíte, že je rýchlejšie jednoducho uložiť vypočítanú hodnotu, ako ju vypočítať za chodu. To platí najmä vtedy, keď používateľ pracuje na starších, pamäťovo obmedzených alebo pomalých počítačoch.
Ďalším bežným dôvodom denormalizácie údajov je poskytnúť možnosť presne reprodukovať dokument tak, ako bol pôvodne vytvorený. Napríklad, ak potrebujete znova vytlačiť faktúru spred roka, ale meno zákazníka sa za posledný rok zmenilo, opätovná tlač faktúry zobrazí nové meno v dokonale normalizovanej databáze. Ak existujú obchodné dôvody, ktoré vyžadujú, aby bola faktúra presne reprodukovateľná, môže byť potrebné, aby bolo meno zákazníka uložené v zázname faktúry v čase vytvorenia faktúry.
Uvedomte si, že väčšina krokov na denormalizáciu databázovej schémy má za následok dodatočný programovací čas potrebný na ochranu údajov a používateľa pred problémami spôsobenými nenormalizovaným návrhom. Napríklad v prípade vypočítaného poľa OrderTotal musíte vložiť kód, ktorý vypočíta a aktualizuje toto pole vždy, keď sa zmenia údaje v poliach tvoriacich základ tejto hodnoty. Toto dodatočné programovanie si, samozrejme, vyžaduje čas na implementáciu a čas na spracovanie za behu.
Nakoniec vždy zdokumentujte všetko, čo ste urobili na denormalizáciu dizajnu. Je úplne možné, že vás alebo niekoho iného zavolajú, aby ste vykonali údržbu alebo pridali do aplikácie nové funkcie. Ak ste zanechali dizajnové prvky, ktoré zdanlivo porušujú pravidlá normalizácie, vašu starostlivo zváženú prácu môže vrátiť iný vývojár v snahe „optimalizovať“ dizajn. Vývojár, ktorý vykonáva údržbu, má samozrejme tie najlepšie úmysly, ale môže neúmyselne obnoviť problém s výkonom, ktorý bol vyriešený jemnou denormalizáciou.
Jedna vec, ktorú treba mať na pamäti, je, že denormalizácia sa takmer vždy vykonáva na účely vykazovania alebo výkonu, a nie len na udržiavanie údajov v tabuľkách. Predstavte si situáciu,

v ktorej bola zákazníkovi poskytnutá špeciálna zľava, ktorá nezodpovedá jeho tradičnej zľave. Môže byť veľmi užitočné uložiť skutočnú sumu fakturovanú zákazníkovi namiesto spoliehania sa na databázu na výpočet zľavy pri každom vytlačení správy. Uloženie skutočnej sumy zaisťuje, že správa vždy odráža sumu fakturovanú zákazníkovi, namiesto vykazovania hodnoty, ktorá závisí od iných polí v databáze, ktoré sa môžu časom meniť.